REMARKS

Claims 1-27 remain pending. Of these, claims 1-12 have been withdrawn from consideration. In the Office Action, the Examiner rejected claims 13-27 under 35 U.S.C. § 102(e) as being anticipated by Dice. (US Patent No. 6,735,760 B1).

Claim objections:

Claim 19 has been amended, as suggested by the Examiner, to delete the repeated "the". Via such amendment, the objection has been overcome.

§ 102(e) rejection:

The Office Action rejects Claims 13-27 under 35 U.S.C. § 102(e) as anticipated by Dice (US Patent No. 6,735,760 B1). However, the Office Action has failed to meet its burden of making a prima facie case of anticipation for the claims, and such rejections should be withdrawn.

Comment: Is this correct?

"[F]or anticipation under 35 U.S.C. 102, the reference must teach *every aspect* of the claimed invention ..." MPEP 706.02 (emphasis added). "The identical invention must be shown *in as complete detail as contained in the ... claim.*" *Richardson v., Suzuki Motor Co.*, 868 F. 2d 1226, 1236, 9 USPQ2d 1913, 1920 (Fed. Cir. 1989) (emphasis added). Dice simply fails to disclose every aspect of the inventions claimed in Claims 13-27. The Examiner has therefore failed to meet his burden of making a prima facie case of anticipation.

Dice discloses the use of a header field to hold a monitor for an object in order to effect synchronization among threads. Dice uses the header/monitor inflation scheme disclosed in Agesen (Ser. No. 09/245,778), but claims that it does not require Agesen's atomic compare-and-swap operations for "meta-lock". Thus, Dice discloses a lock-based synchronization scheme.

In contrast, Applicants do not claim a lock-based scheme. Indeed, Applicants expressly teach away from lock-based approaches in paragraph 4 of the Application:

> **[0004]** Such locking synchronization mechanisms, however, may require a substantial amount of overhead for associated system calls (e.g., for keeping track of which threads own which locks and/or the states of locks). This overhead may be large enough to impact performance if the locks are used in performance-critical code. Also, these locking mechanisms may not be tolerant of faults. For example, if a thread that has acquired a lock dies (e.g., is killed or exits abnormally) before releasing its lock, other threads may be prevented from accessing the shared resources protected by that unreleased lock. Further, if a thread that owns a lock is suspended, a "deadlock" may occur if the resumption of the suspended thread is dependent on the completion of another task whose progress, in turn, is dependent on the lock of the suspended thread.

Thus, Dice and Applicants have approached thread synchronization in vastly different manners. Dice does not disclose, suggest or teach Applicants' claimed invention.

Regarding Claim 13, the claim recites, in part, "performing an atomic store operation to add the element to the data structure ..." The Office Action claims that this element is shown at Col. 12, ll. 15-27 and Col. 14, ll. 17 – 26 of Dice. This is not correct. These portions of Dice show, respectively, writing a value to an object's existing lock word and maintaining monitor (e.g., lock) records for uncontested locking (e.g., locked object is accessed by the same thread). The first cited section (Col. 12, ll. 15-27), simply indicates that the lock itself is written. There is no indication whatsoever that an element is added to a data structure. The second cited section (Col. 14, ll. 17 – 26) returns a monitor record to a free list for uncontested locking. This does not read on Applicant's claim. **Modifying a lock by placing a value into an existing field is not**

equivalent to "add"-ing an element to a data structure. Similarly, **freeing a lock record for uncontested (single-thread) locking is not equivalent to "add"-ing an element to a data structure.** The Office Action claims that the cited sections of Dice show "threads attempt to access an object". However, this is not sufficient to make a prima facie case of the claimed elements. **Threads attempting to access an object is not equivalent to "an atomic store operation to add the element to the data structure"**. For at least the foregoing reasons, the Office Action has failed to make a prima facie case of anticipation with respect to Claim 13.

Claim 13 also recites, in part, "determining via the first thread whether an element is a member of a data structure". The Office Action claims that this element is shown at Col. 12, ll. 15-28 and Col. 14, ll. 27-52 of Dice. This is not correct. As is explained immediately above, the first cited section of Dice (Col. 12, ll. 15-28) shows that a value is written to an existing lock word. Placing a value into an existing lock word, as is disclosed by Dice at Col. 12, ll. 15-28, is not equivalent to determining whether a first thread is an element of a data structure. Admittedly, Dice discloses a CAS operation whereby the contents of the lock word are read before they are written. Still this does not read on or suggest the claim language of Claim 13. **Reading a value of a lock is not equivalent to determining whether an element is a member of a data structure**. Dice reads a value to determine whether an object is locked or not; it does not disclosure determining whether an element is actually in the data structure or not. This is woefully insufficient not make a prima facie case of anticipation regarding Claim 13.

In the latter cited section (Col. 14, ll. 17 – 26) of Dice, a second thread reads the lock to determine that the object is owned by another thread. Again, **reading a value of a lock is not**

equivalent to determining whether an element is a member of a data structure. Whether an item is locked or not tells you nothing about what elements are a member of the item.

Accordingly, the rejection of Claim 13 must fail because the Office Action has failed to make a prima facie case of anticipation over Dice, at least for one or more of the reasons set forth above. In addition, Claims 14-20, which depend from Claim 13, are also allowable.

Also, the rejection of Claim 21 must fail because the Office Action has failed to make a prima facie case of anticipation over Dice. For instance, Claim 21 recites, in part, "one or more program instructions cause a branch to be taken in response to determining that a pointer value for an element is not equal to an initialization value." Claim 21 further teaches that the branch instruction is to "branch around one or more atomic operations to add the element to the data structure, such that the atomic operations are *not* executed" (emphasis added). Dice does not teach, disclose, nor suggest a branch instruction. Also, the cited portion of Dice does not teach a branch instruction to be taken so that elements are NOT added if the pointer value is not equal to the initialization value. Instead, Dice teaches that monitor records *are* added if the list of monitor records for a thread has become empty. A prima facie case of anticipation thus has not been made with respect to Claim 21. Claim 21 is therefore allowable for at least one or more of the reasons set forth above. In addition, Claims 22-27, which depend from Claim 21, are also allowable.

Applicant reserves the right to argue other assertions made in rejecting these claims in the future. Therefore as argued above, the rejections of all claims pending in the application, i.e. claims 13-27, should be withdrawn and the claims allowed.

The Examiner is welcome to contact the Attorney of Record, Shireen Irani Bacon (Reg. No. 40,494) at 512.263.1250 to discuss any matters with the case. The Commissioner is hereby authorized to charge any fees in connection with this communication to our Deposit Account No. 50-0221.

To the extent necessary, a petition for an extension of time under 37 C.F.R. § 1.136 is hereby made. Please charge any shortage in fees due in connection with the filing of this paper, including extension of time fees, to Deposit Account No. 50-0221 and please credit any excess fees to such deposit account.

Respectfully submitted,

Dated: June 30, 2008

/Shireen Irani Bacon /
Shireen Irani Bacon
Registration No. 40,494

Intel Corporation
Customer Number 59796
c/o Intellevate, LLC
P.O. Box 52050
Minneapolis, MN 55402